

Visualization of Contributions to Open-Source Projects

Andreas Schreiber
German Aerospace Center (DLR),
Institute for Software Technology
Köln, Germany
andreas.schreiber@dlr.de

ABSTRACT

We analyze visually, to what extent team members and external developers contribute to open-source projects to have a high-level impression about collaboration in that projects. For that, we record the provenance of the development process and draw the resulting property graph. Our graph drawings show, which developers are jointly changed the same files what we apply to Germany's COVID-19 exposure notification app 'Corona-Warn-App'.

CCS CONCEPTS

• **Human-centered computing** → **Graph drawings**; • **Software and its engineering** → **Open source model**; **Programming teams**.

KEYWORDS

graph visualization, software visualization, provenance, open source

ACM Reference Format:

Andreas Schreiber. 2020. Visualization of Contributions to Open-Source Projects. In *The 13th International Symposium on Visual Information Communication and Interaction (VINCI 2020)*, December 8–10, 2020, Eindhoven, Netherlands. ACM, New York, NY, USA, 2 pages. <https://doi.org/10.1145/3430036.3430057>

1 INTRODUCTION

A transparent and traceable team composition and development process is one of the advantages of the open-source model. Understanding the characteristics of open-source projects, where developers with different roles work together, is an important question for projects with high public interests.

The COVID-19 pandemic raises challenges for software developers to help to fight the pandemic with software systems, which must be developed under time pressure. For example, mobile apps for *contact tracing* of infected persons such as Germany's exposure notification app *Corona-Warn-App* (CWA).

To analyze, to what extent team members and external contributors contributed to CWA on GitHub, we record and store the *provenance of the software development process* [8] according to the provenance data model PROV [5] in a graph database (Section 2). We query a sub-graph for visualization with graph drawing (Section 3)—which we apply to the CWA (Section 4).

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).

VINCI 2020, December 8–10, 2020, Eindhoven, Netherlands

© 2020 Copyright held by the owner/author(s).

ACM ISBN 978-1-4503-8750-7/20/12.

<https://doi.org/10.1145/3430036.3430057>

2 PROVENANCE OF SOFTWARE DEVELOPMENT PROCESSES

We extract provenance of a software development process from repositories with Git2PROV [2, 7] and store it as a property graph in the graph database Neo4j for further analysis [6].

We use the graph to answer the question: “Which files have commits by team members as well as external contributors?” by adding information about the contributors roles as additional edges using database queries with CYPHER; the role of each contributor, which is necessary to construct the proper CYPHER, is requested from GitHub via their API. We make the assumption, that members of the GitHub organization corona-warn-app are dedicated team members. Then we query for files, where team members and external contributor made changes at any of the files revisions and visualize the result (Figure 1; see Section 3).

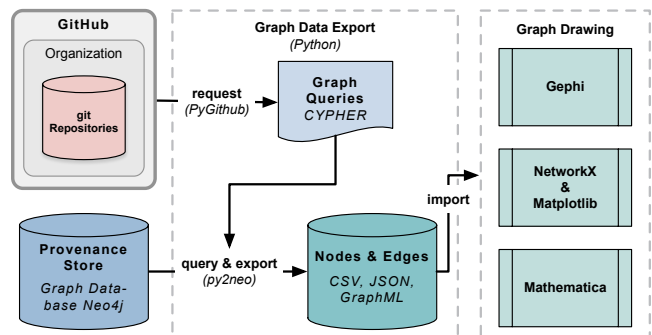


Figure 1: Querying and exporting graph data for visualization.

3 GRAPH VISUALIZATION

We visualize parts of the property graph with GEPHI [1]. During querying and exporting for visualization, we map the property graph as follows: files (i.e., PROV entities) and contributors (i.e., PROV agents) become graph nodes with two distinct colors and contributions become edges, which color depends on their property role (i.e., “team member” or “external contributor”).

For the *coloring*, we choose distinct colors from two different qualitative color schemes generated by COLORBREWER [3]. The *size of nodes* are proportional to their degree. In our current approach, we generate two drawings for each project; one where we scale the node sizes according to the *in-degree* of file nodes and a second one where we scale according to the *out-degree* of contributors. For the *layout* we use the ForceAtlas2 algorithm [4].

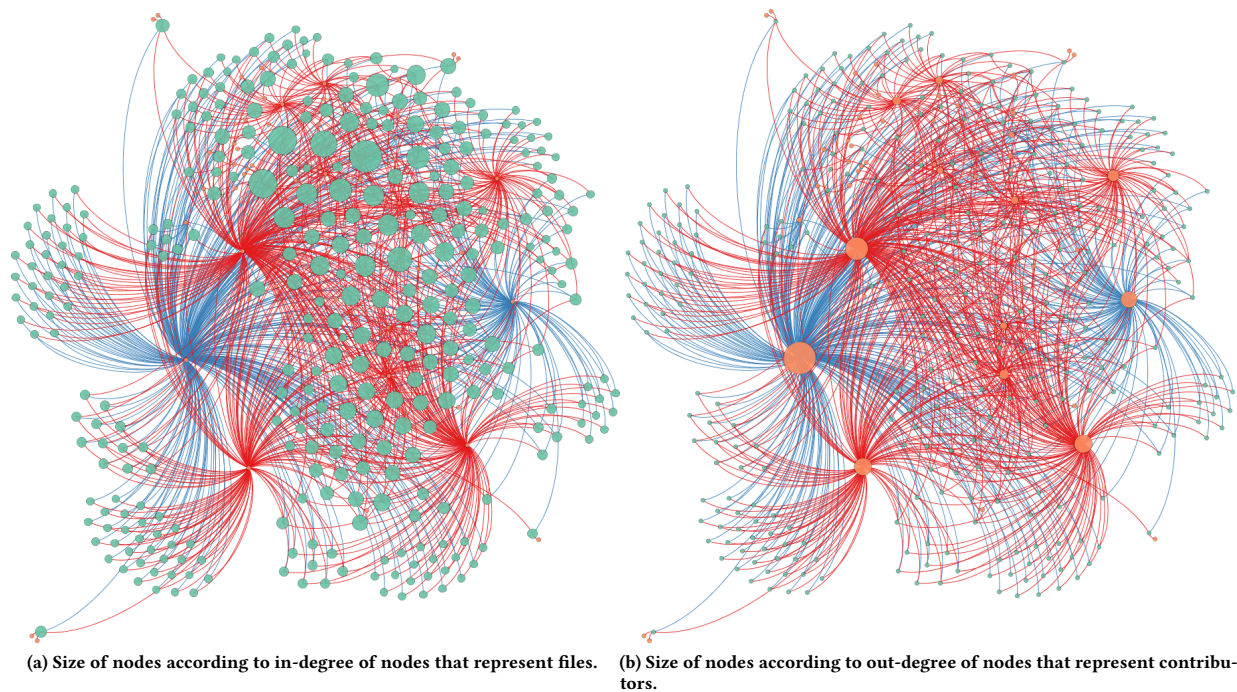


Figure 2: Files (●) and contributors (●) for the cwa-app-android project. Red edges indicate file changes by team members (●→●). Blue edges indicate file changes by external contributors (●→●).

4 USE CASE: CORONA-WARN-APP (CWA)

The CWA has been developed in a short time frame using an open development process—publicly available from 12 repositories¹: development started in April 2020 and the app was released on 16th June, 2020 for Android and iOS.

We visualize the Android app cwa-app-android, for which the database has 3672 Entities, 56 Agents, and 379 Activities. This leads to 571 contributions by team members and 1230 contributions by external contributors. We draw one graph where we scale node sizes proportional to the *in-degree* of file nodes (Figure 2a) and a second one where we scale proportional to the *out-degree* of contributors (Figure 2b).

Our graph drawing show typical patterns: team members and external contributors work collaboratively on many files. Because the drawing is based on provenance data, the interpretation is that over the time of development many files were changed by developers with different roles; where a small numbers of developers made most of the changes.

5 CONCLUSION AND FUTURE WORK

We presented graph drawings to visually see patterns how team members and external contributor worked on the same files in open-source projects.

Future work includes user studies to evaluate readability and faithfulness. The graph drawings surely can be improved in many

ways, for example, with other layouts, color schemes (especially to support color blindness), transparency, or shapes.

We add additional provenance information besides the files operations in GIT—for example, actions from the GitHub platform such as issues changes, pull requests, documentation changes, and *GitHub Actions*. The knowledge within this provenance information allow a much broader set of questions beside file changes.

REFERENCES

- [1] Mathieu Bastian, Sebastien Heymann, and Mathieu Jacomy. 2009. Gephi: An Open Source Software for Exploring and Manipulating Networks. <http://www.aiai.org/ocs/index.php/ICWSM/09/paper/view/154>
- [2] Tom De Nies, Sara Magliacane, Ruben Verborgh, Sam Coppens, Paul Groth, Erik Mannens, and Rik Van De Walle. 2013. Git2PROV: Exposing Version Control System Content as W3C PROV. In *Proceedings of the 12th International Semantic Web Conference – Volume 1035 (ISWC-PD '13)*. 125–128.
- [3] Mark Harrower and Cynthia A. Brewer. 2003. ColorBrewer.org: An Online Tool for Selecting Colour Schemes for Maps. *The Cartographic Journal* 40, 1 (2003), 27–37. <https://doi.org/10.1179/000870403235002042>
- [4] Mathieu Jacomy, Tommaso Venturini, Sebastien Heymann, and Mathieu Bastian. 2014. ForceAtlas2, a Continuous Graph Layout Algorithm for Handy Network Visualization Designed for the Gephi Software. *PLOS ONE* 9, 6 (06 2014), 1–12. <https://doi.org/10.1371/journal.pone.0098679>
- [5] Luc Moreau and Paul T. Groth. 2013. *Provenance: An Introduction to PROV*. Morgan & Claypool Publishers. <https://doi.org/10.2200/S00528ED1V01Y201308WBE007>
- [6] Andreas Schreiber and Claas de Boer. 2020. Modelling Knowledge about Software Processes using Provenance Graphs and its Application to Git-based Version Control Systems. In *42nd International Conference on Software Engineering Workshops*. IEEE/ACM, Seoul, Republic of Korea.
- [7] Ruben Verborgh, Sara Magliacane, Andreas Schreiber, and Vlad Korolev. 2020. *onyame/Git2PROV: Improved error handling*. <https://doi.org/10.5281/zenodo.3942169>
- [8] Heinrich Wendel, Markus Kunde, and Andreas Schreiber. 2010. Provenance of Software Development Processes. In *Provenance and Annotation of Data and Processes*. Springer Berlin Heidelberg, 59–63.

¹<https://github.com/corona-warn-app>